Developer Guide

Introduction

This *Developers Guide* is intended to drive you along the building process of all binary modules of the eXtraWay Platform.

The source code will produce 32 bit binaries. Porting to 64 bit will follow.

> Since the code is the same for both Windows and Unix/Linux compiling, please follow the proper instruction.

The documentation refers to experienced developing environments. Developers are free to experiment other environments but pay attention to unattended behaviour that may occur.

First of all the developer that approaches to eXtraWay Platform should take care of downloading the proper source code from the given SVN repository after required registration is done.

The code appears as a sequence of directories containing source code, documentation, instructions and compiler driven solutions. Let's take a look to some of them

- 3rdp: directory for 3rd party code used within `eXtraWay Platform`. See 3rd party chapter for details.
- docs: contains some basic documentations and will be used to generate the *Developers Reference*;
- redist: contains all binaries and configuration needed to build up the first *eXtraWay Platform* installation;
- solutions: contains Visual Studio projects and Unix/Linux Makefiles.

Following chapter will drive you to configure the required environment to build *eXtraWay Platform* modules.

Common Configuration

Common configuration concerns the basic requirements that are:

- Access SVN repository;
- Generate code documentation

In both cases developer can operate using thier own tools but we now declare how to act using Eclipse Indigo CDT.
Download and install the suitable version for your Operating System, in our example the 8.0.2 SR2 version, than install remaining parts as follows.
Access menu `Help → Install New Software…` and add the SVN plug-in. We use *Subversive* from http://community.polarion.com/projects/subversive/download/eclipse/2.0/indigo-sr2-site/[1].
The package contains many parts, the suggested ones are:

- Subversive SVN Connector;
- SVNKit 1.3.5 Implementation;
- Native JavaHL 1.6 Implementation.

We also suggest to update SVNKit to last available version at `http://eclipse.svnkit.com/1.3.x`

Access to SVN repository

Once the developer has the SVN Url and credential[2], he can access the repository and `Check Out` all modules and files from SVN `ROOT → trunk`.
Than a `libs` directory should be created within the project. That directory will contain every `[.lib|.a]` files after compiling procedure.

> During following development activities please take care to ignore every file that will be built into directories named 'libs', 'od' or 'or' inside source files directories. Those files must never get back to SVN.

Third Party

The `eXtraWay Platform` uses third party code. That code belongs to he Open Source World and most of it is part of every `Unix/Linux` environment.
Unfortunatelly we can't say the same for `Windows` where libraries as `libxml2` are not natively distributed with O.S.
To find a valid workaround to this problem, in order to mantain code as much similar as possible between Windows and Unix/Linux development, we searched for the equivalent Windows code that is actually stored in `3rdp` directory.

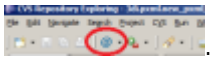
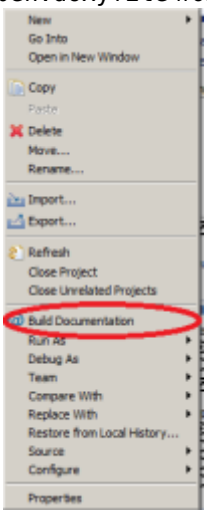
> **That code is not from 3D Informatica but comes from their authors. We just distribute the code in order to use properly include files.**
> **`3rdp` directory also includes a `setup` directory where you can find the *Dynamic Linked Libraries* ready to use in Windows accompained by their license disclaimer and information concerning where and how to get the original code.**

Generate code documentation

The whole code is properly commented in order to use Doxygen to generate the source code *Reference*.
Since the documentation was originally written in Italian and than translated, many parts could be incomplete or incorrect. Please tell us which documentation requires our attention.

First of all Donwload and Install a suitable version of Doxygen. We used 1.8.0 but even previous version should operate properly.

Now complete Eclipse setup adding Doxygen Plug-in as follows:
Access menù Help → Install New Software… and add the *Eclox* plug-in from http://download.gna.org/eclox/update. Select all package parts.

After the setup end a new button appears:  .
Now you can build documentation by selecting the xw Open.doxyfile from docs directory and push that button or require the documentation generation from context menu as follows 

P.S.: *Eclox* plug-in should be configured to identify where Doxygen resides on you computer.

Windows Development

In order to build Win32 modules version you need to install the following software:

- Microsoft Visual C++ 2008 Express Edition SP1;
- Microsoft Windows SDK for Visual Studio 2008 SP1 Express Tools for Win32.

As told before, developers are free to adopt a different compiler but all provided material is verified in a similar environment.

Now open you *Microsoft Visual C++ 2008* and open the solution in solution\eXtraWay Platform.sln.
Set the xw project as primary one (project dependecies should be already set).
Chose the required configuration between Debug & Release and simply compile.
The result will produce:

- Debug Confuguration:
  - a directory named od in each source dir containing .obj and other files;
  - one or more <libname**d**>.lib file into libs directory;
  - a directory named bin_d9 near to the project that will contain the produced binaries.
- Release Configuration:
  - a directory named or in each source dir containing .obj and other files;
  - one or more <dirname>.lib file into libs directory;
  - a directory named bin9 near to the project that will contain the produced binaries.

Windows modules also uses 3rd party code that isn't commonly distributed on Windows. We're talking about libraries as:

- libxml2;
- libxslt;
- libiconv;
- zlib;
- libzip;
- openssl… and others.

In order to work properly all this code, belonging to the Open Source World, must be set up before you try to execute modules from *eXtraWay Platform*.
Please go to 3rdp\setup directory and follow instruction into README.TXT file to generate the xw3rdp-setup.exe file.

You will need WinRAR or you can build your own set-up procedure.

Some other steps must be done, please see the "Completing the Environment" chapter.

Unix/Linux Development

In order to build Unix/Linux modules version you need to install the following software:

- build-essential;
- GCC version 4.1.2 and proper g++ package;
- libxml2-dev;

- libxslt-dev;
- libssl-dev;

As told before, developers are free to adopt a different compiler but all provided material is verified in a similar environment.

Now go to `solution` directory and simply run `make` using the local `Makefile`.
The following parameters can be used:

- 'free': all intermediate and final files will be removed and following build will occur from scratch;
- 'deb' or nothign as defult: the etire project will be built in `Debug` mode;
- 'rel': the entire project will be built in `Release` mode;
- 'out': all the output produced from single build processes will feed a single couple of files: `make.outd` for `Debug` configuration and `make.outr` for `Release` configuration

Please run a `make free` before any other make command in order to create intermediate files output directories.

Since we use to build for different platform and in different configuration (older gcc, various glibc version and so on), the result will generate separate directory for each `hostname`.
Say that you computer's name is `develXw` the result will produce:

- `Debug` Configuration: a directory named `develXw/od` in each source dir containing .o and other files. Libraries (.a) and executable will built into the same directories;
- `Release` Configuration: a directory named `develXw/or` in each source dir containing .o and other files. Libraries (.a) and executable will built into the same directories.

Changing Environment

Each module that composes `eXtraWay Platform` has it's own `Makefile`. The `Makefile` in `solutions` directory just execute `make` for each module.

There's an interesting file into the `include` directory that the developer could examine: `macros.mkh`.
This file is used by each `Makefile` and try to adapt the building procedure to the environment.
We've experienced compiling the module on variuos platforms (Sun Solaris, Linux, Aix). The developers are invited to follow this style in order to let the modules' `Makefile` as unchanged as possibile.

Completing the Environment

Now that the binaries have been compiled succesfully, let's see how those files should be used to set up an *eXtraWay Platform simple installation*.

Take a look to `redist` directory. It contains some other directories with the minimum amount of files you need to set up the whole environment[3)].

- `bin` directory contains a few files. Put here the `xw3rdp-setup.exe`. You can run it before running `eXtraWay Server` or just run the server that will check for library presence and run that setup if needed.
  While `msg.dat` must be used both in Windows and Linux installation there are some differences between the Operating Systems:
  - `vcredist_x86.exe` and `vcredist_x64.exe` are needed in proper Windows O.S. in order to provide the computer of required run-time libreries;
  - `xw.rc` is required only in `Unix/Linux` installations.
- `conf` directory contains the basic configuration files. The content of this file is documented into eXtraWay common documentation available at 3D Informatica documentation site;
- `context` directory will bring some information concerning the environmnet in which archives are generated and handled in order to mark each *Information Unit* with environment meta data;
- `logs` directory will contain each kind of log the modules will produce, please refer to Logs configuration and documentation for details. Natively empty;
- wd directory will be used from `WatchDoc` procedure to import data from XML files. Natively empty.

The next step to do is to copy binary files(xw[.exe], xwls[.exe] and shared libraries (libxwwd[.dll|.so]) from thery compiler directory into the `bin` directory than we're ready to start.

Starting eXtraWay in Windows

You can act in two different ways:

- Simply esecute eXtraWay Server (xw.exe) from a simple command line. The server will verify if it's installed as a service or not so it can take up to 20/30“ to start. Use the `-noserv` parameter to declare it's been run manually and reduce startup time.
  You can also use the `-p<port>` parameter to force eXtraWay starting on a socket port different from the default that is **4859**;
- Install *eXtraWay Server* as a Winodws Service. Use the command line parameter `-service_install` to install and start the service, and `-service_uninstall` to stop and uninstall the service. You can also use `-service_start` & -

`service_stop` to governate the service.

If `eXtraWay Server` is started manually you'll see it's icon into the *try bar*. You can click on it and view version and other informations. You can also register the server in order to complete `context` set-up. This step is required in order to insert and modify records containing `context` meta-data.

Starting eXtraWay in Unix/Linux

“Ça va sans dire”: pay attention to user and rights of the provided files. We're used to choose an `extraway` user and use it to develop and install packages.

The Unix/Linix distribution is accompanied by a `xwclt` script. As any other Service script it supports the following commands:

- start
- stop
- restart
- status

You can *simlink* this script into your `/etc/init.d` directory in order to set and use `eXtraWay Server` as a service.

Other commands are also availbale:

- vstatus : provides a verbose status telling the Process ID of each module.
- version : provides the version list for all the modules.

The `xwctl` script uses `hwadmin` to send the server specific commands[4] and the `../conf/xwctl.conf` file to choose default startup parameters.
If you need to run manualy the server you can specify the `-p<port>` parameter do set the socket port to use. Default is **4859**.

In order to register the server and complete the `context` set-up, needed to insert and modify records with `context` meta-data, please run `./xw -r` and follow the instruction.

Registering the Server

The registration process is quite simple and only requires 3 values:

- A serial number: normaly used by 3D Informatica to identify each installation and identify the environment;
- A User Name;
- A Compamy/Organization Name.

After the registration ends the server will close and must be run again.

eXtraWay Server Managment

Now the server has been compiled and is properly running, you have to build the console to use it.
This procedure will also give you the source code of the main API required to converse with eXtraWay Server, that means, the source code of the `eXtraWay Broker`.

Following the same steps shown before, checkout from SVN repository the `xwbroker` project from the main `trunk` than follow these steps:

- Go to the project properties and convert the project to faceted form.
- Go to the project properties, choose the `Java Build Path` menù than the `Liraries` tag. Now `Add JARs…` and the add following libraries from `console\xway\WEB-INF\lib` :
  - activation.jar
  - common.jar
  - dom4j.jar
  - jaxen.jar
  - log4j.jar
  - mail.jar
  - saxon*.jar
  - xalan.jar
  - xercexImpl.jar
- From the same panel 'Add External JARs… and choose *servlet-api.jar* from your Tomcat installation libraries. Ok, now it's time to build your project. Only warnings should be present. The apps.jar and hj.jar libraries will be built into the same console\xway\WEB-INF\lib directory. Now copy the console\xway\xway.xml file into your Tomcat\conf\Catalina\localhost'' directory and edit it in order to set the correct directory where the application resides.

Good… now your eXtraWay Console will answer to http://localhost:8080/xway/engine/console.jsp

1)
Pay attention to the url that refers to SR2 version
2)
i.e.: something as svn+ssh://devname@host.foo.bar/var/opt/svn/xwopen

3)

Complete installation is richer

4)

i.e. to stop the server